A Swarm with an Effective Information Sharing Mechanism for Unconstrained and Constrained Single Objective Optimization Problems

Tapabrata Ray
Center for Advanced Numerical Engineering Simulations,
School of Mechanical Engineering,
Nanyang Technological University,
Singapore 639798.
Email: mtray@ntu.edu.sg

K. M. Liew
Center for Advanced Numerical Engineering Simulations,
School of Mechanical Engineering,
Nanyang Technological University,
Singapore 639798.
Email: mkmliew@ntu.edu.sg

Abstract- In this paper we present an effective multilevel information sharing strategy within a swarm to handle single objective, constrained and unconstrained optimization problems. A swarm is considered as a collection of individuals having a common goal to reach the best value (minimum or maximum) of a function. The success of a swarm is attributed to the identification of a set of competent leaders and a meaningful information sharing scheme between the leaders and the rest of the individuals that enables the swarm to collectively attain the common goal. The proposed algorithm mimics the above behavioral processes of a real swarm and maintains unique individuals at all time instants. The uniqueness among the individuals result in a set of near optimal solutions at the final phase that is useful for sensitivity analysis. The benefits of the effective information sharing strategy is illustrated by solving two unconstrained problems with multiple equal and unequal optima and a constrained optimization problem.

1 Introduction

Swarm strategies are fairly new methods that have received considerable attention over recent years as optimization methods for complex functions. Kennedy and Eberhart (1995) (1997) initially proposed the swarm strategy for optimization. Unlike conventional evolutionary methods, the swarm strategy is based on simulation of social behavior where each individual in a swarm adjusts its *flying* according to its own *flying* experience and companions' *flying* experience. The key to the success of such a strategy in solving an optimization problem lies with the mechanism of effective information sharing across individuals. After the initial concept was proposed, there had been comparative studies between swarm and other evolutionary strategies by Eberhart and Shi (1998) and Angeline (1998).

It is well known that the presence of constraints significantly affects the performance of all optimization algorithms including evolutionary search methods. There has been a number of approaches to handle constraints in the domain of evolutionary methods including rejection of infeasible individuals, penalty functions and their variants, repair methods, use of decoders, separate treatment of constraints and objectives and hybrid methods incorporating knowledge of constraint satisfaction. Michalewicz (1995) provides a comprehensive review on constraint handling methods. All the methods have limited success as they are problem dependent and require a number of additional inputs. Penalty functions using static, dynamic or adaptive concepts have been developed over the years. All of them suffer from common problems of aggregation and scaling. Repair methods are based on additional function evaluations, while the decoders and special operators or constraint satisfaction methods are problem specific and cannot be used to model a generic constraint. Separate treatment of constraints and objectives is an interesting concept that eliminates the problem of scaling and aggregation.

Constraint handling using a Pareto ranking scheme is a relatively new concept having its origin in multiobjective optimization. Fonseca and Flemming (1995) proposed the Multiple Objective Genetic Algorithm (MOGA), a Pareto ranking scheme to handle multiple objectives while the Nondominated Sorting Genetic Algorithm (NSGA) was introduced by Srinivas and Deb (1994) for multiobjective problems. Ray et al. (2000) introduced a multiobjective evolutionary algorithm that is based on Pareto ranking in both the objective and constraint domain suitable for a wide variety of single or multiobjective constraint problems.

For a swarm strategy to be efficient for constrained problems there is additional information about the constraint satisfaction that needs to be meaningfully shared among the individuals. The information sharing strategy proposed in this paper relies on Pareto methods to handle constraints. The strategy also ensures that distinct points are maintained throughout the search, as such points are useful for sensitivity analysis. The proposed multilevel information sharing strategy is described in detail in the next section separately for unconstrained and constrained problems.

2 Mathematical Formulation

A general constrained optimization problem (in minimization sense) is presented as:

0-7803-6657-3/01/$10.00 ©2001 IEEE

Minimize $f(x)$

subject to $g_i(\mathbf{x}) \geq a_i \ , i = 1,2,\ldots,q$

$$h_j(\mathbf{x}) = b_j \ , j = 1,2,\ldots,r$$

where there are q inequality and r equality constraints, $\mathbf{x} = [x_1 \quad x_2 \quad \ldots \quad x_n]$ is the vector of n design variables and a_i and b_j are constants.

It is a common practice to transform the equality constraints (with a tolerance δ) to a set of inequalities and use a unified formulation for all constraints.

$h_j(\mathbf{x}) \leq b_j + \delta$ which is same as $-h_j(\mathbf{x}) \geq -b_j - \delta$ and $h_j(\mathbf{x}) \geq b_j - \delta$. Thus r equality constraints will give rise to $2r$ inequalities, and the total number of inequalities for the problem is denoted by s, where $s=q+2r$. For each individual, $\mathbf{c}$ denotes the constraint satisfaction vector given by $\mathbf{c} = [c_1 \quad c_2 \quad \ldots \quad c_s]$ where

$$c_i = \begin{cases} 0 & \text{if } i^{th} \text{ const. sat., } i = 1,2,\ldots,s \\ a_i - g_i(\mathbf{x}) & \text{if } i^{th} \text{ const. vio., } i = 1,2,\ldots,q \\ b_i - \delta - h_i(\mathbf{x}) & \text{if } i^{th} \text{ const. vio., } i = q+1, q+2,\ldots,q+r \\ -b_i - \delta + h_i(\mathbf{x}) & \text{if } i^{th} \text{ const. vio., } i = q+r+1, q+r+2,\ldots,s \end{cases}$$

For the above c_i's, $c_i = 0$ indicates the i^{th} constraint is satisfied, whereas $c_i > 0$ indicates the violation of the constraint. The **CONSTRAINT** matrix for a swarm of M individuals assumes the form

$$\textbf{CONSTRAINT} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1s} \\ c_{21} & c_{22} & \cdots & c_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ c_{M1} & c_{M2} & \cdots & c_{Ms} \end{bmatrix}$$

In a swarm of M individuals, all nondominated individuals based on the constraint matrix are assigned a rank of 1. The rank 1 individuals are removed from the swarm and the new set of nondominated individuals is assigned a rank of 2. The process is continued until every individual in the swarm is assigned a rank. Rank=1 in the constraint matrix indicate that the individual is nondominated. It can be observed from the constraint matrix that when all the individuals in the swarm are infeasible, the nondominated ones will have a rank of 1. Whenever there is one or more feasible individual in the swarm, the feasible solutions will take over as rank 1.

The initial swarm consists of a collection of random individuals. Over time, the individuals communicate with their neighboring better performers and derive information from them to look for optimum in smaller groups. The search strategy is schematically shown in Figure 1. The larger dots indicate the leaders communicating with their neighboring individuals (smaller dots) in quest for optima. All members that have the same leader form a group.

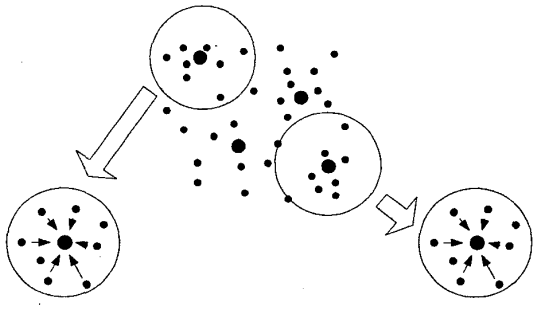

Figure 1. Interaction among Individuals and their Group Leader

Having described briefly the general formulation of the constraint matrix and the concept of Pareto ranking, the pseudo code of the algorithm is presented.

2.1 Algorithm

Initialize M unique individuals in the Swarm;
Unconstrained Problem Strategy
Do {
    Compute Objective Values of each Individual in the Swarm;
    Compute Average Objective Value for the Swarm;
    For (each Individual) {
        If (Objective Value of an Individual ≤ Average Objective Value for the Swarm) : Assign Individual to Better Performer List (BPL);
    }
    For (each Individual not in BPL) {
        Do {
            Select a Leader from BPL to derive information;
            Acquire information from the Leader and move to a new point in the search space;
        } while (all individuals are non unique)
    }
} while (termination condition = false)

Constrained Problem Strategy
Do {
    Compute Objective values for each Individual in the Swarm;
    Compute Constraint values for each Individual in the Swarm;
    Use Non Dominated Sorting to Rank Individuals based on Constraint Matrix;

```
Assign all Individuals with Constraint Rank = 1 to BPL;
If (size of BPL > M/2) {
        Compute Average Objective Value for the
        Individuals in the BPL;
        Shrink the BPL to contain Individuals with
        Objective value ≤ Average Objective Value;
}
For (each Individual not in BPL) {
        Do {
            Select a Leader from the BPL to derive
            information;
            Acquire information from the Leader and
            move to a new point in the search space;
        } while (all individuals are not unique)
}
} while (termination condition = false)
```

For a constrained problem, individuals with a constraint rank = 1 (nondominated based on constraint matrix) are the best performers based on constraint satisfaction. When there are no feasible individuals in the swarm, the BPL consists of all nondominated individuals based on the constraint matrix. When there is one or more feasible individual, the BPL will consist of all feasible individuals as they will have a constraint rank = 1. The size of BPL is expected to grow, as more and more individuals become feasible. If the size of this list grows more than half the swarm size (i.e. half of the swarm consists of feasible individuals), individuals are allowed to be members of BPL only if they are better than average performers based on objective values. This process ensures that there is a pressure maintained among the feasible BPL members to improve their objective performance to remain as BPL members.

2.2 Selection of Leader

An individual identifies its closest BPL member in the variable space as its leader to derive information. The process is similar as in a real swarm where an individual adjusts its *flying* based on information derived from its better performing neighbors.

2.3 Acquiring Information through Generational Operator

A simple generational operator is used to acquire information from a leader. The operator can result in a variable value even if it does not exist in either the individual or its leader, which is useful to avoid premature convergence. The probability of a variable value generated between an individual and its leader is 50%. The probability of a variable value generated between the lowerbound of the variable and the minimum among the individual and its leader or between the upperbound of the variable and the maximum among the individual and its leader is 25% each. The same generational operator has been used within an evolutionary algorithm in Ray et a. (2000) and also within a swarm in Ray et al.(2001). Other operators that generate intermediate variable values like the blend crossover (BLX) as proposed by Eshelman and Shaffer (1993) or the simulated binary crossover (SBX) as proposed by Deb and Agrawal (1995) can also be used instead.

3 Examples

The performance of the algorithm is illustrated by the following examples. The first two examples are typically interesting as they illustrate the behavior of swarm and highlight its capability in finding multiple optima simultaneously. The next problem illustrates the applicability of the algorithm to solve constrained optimization problems.

3.1 Equal Optimum Problem

This example is obtained from Goldberg and Richardson (1987). It is a single variable problem that has five equal optimums with $0.0 \leq x \leq 1.0$.. The minimization of the function is carried out using the proposed strategy.

Minimize $f(x) = -1.0 * \sin^6(5.1\pi x + 0.5)$

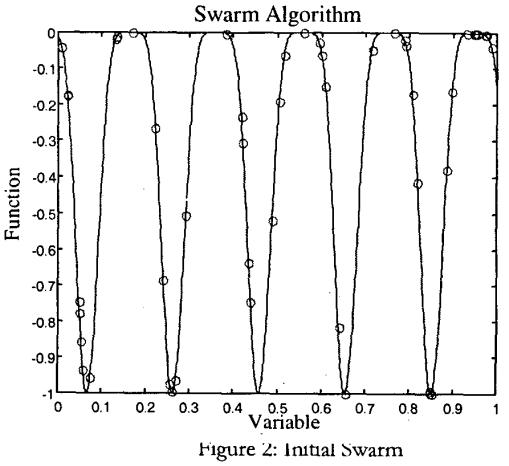

Figure 2: Initial Swarm

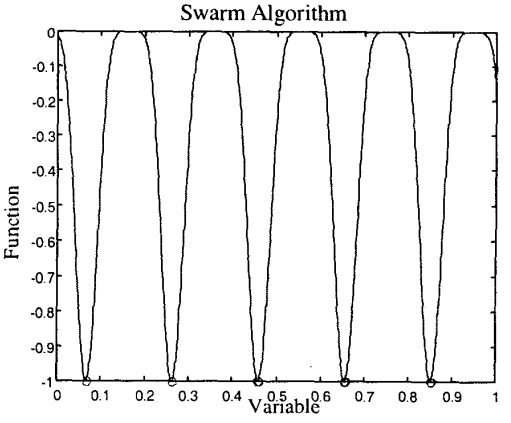

Figure 3: Final Top 50% of Swarm

The initial swarm is presented in Figure 2 while the top 50% of the final swarm is presented in Figure 3. The swarm consists of 50 individuals flying for 100 time steps.

Figure 4 shows a distribution of near optimal individuals forming smaller subgroups of 2 to 7 individuals near each optimum with functional values varying between -0.9991 and -1.000. It can be observed from Figure 4 that the individuals arrived in groups of 2 to 7 to all the five optima. The algorithm performed 2,550 function evaluations and required 0.71s of CPU time on an Origin 2000 system.

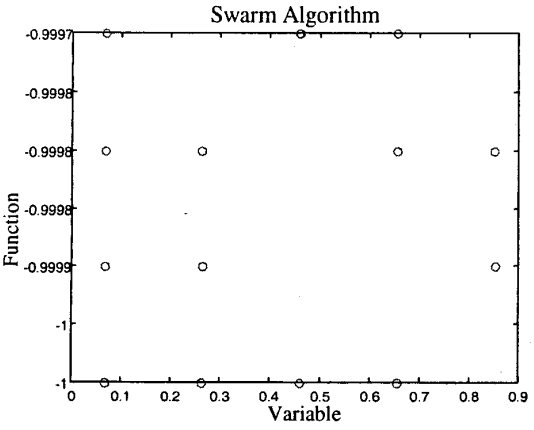

Figure 4: Final Top 50% of Swarm

3.2 Unequal Optimum Problem

This example is also taken from Goldberg and Richardson (1987). It is a single variable problem that has five unequal optimuma with $0.0 \leq x \leq 1.0$.. The minimization of the function is carried out using the proposed strategy with a swarm size of 50 and for 100 time steps.

Minimize

$$f(x) = -1.0 * \sin^6(5.1\pi x + 0.5)\exp\left(\frac{-4\ln(2)(x-0.0667)^2}{0.8^2}\right)$$

The initial swarm is presented in Figure 5. The top 50% of the final swarm is presented in Figure 6 that shows the distribution of near optimal points. It can be observed from Figure 6 that all the BPL members have reached the global optimum. The algorithm performed 2,550 function evaluations and required 0.70s of CPU time on an Origin 2000 system. The sensitivity i.e. the variance and the mean under assumed variations of the variable around the global optimum can also be computed from the top 50% swarm information.

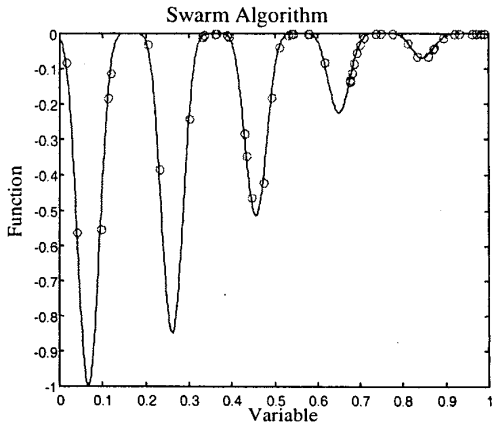

Figure 5: Initial Swarm

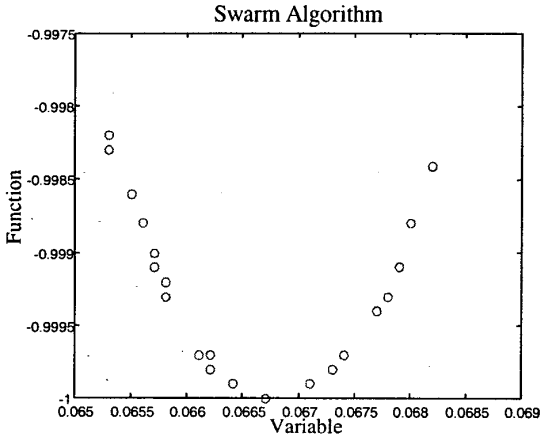

Figure 6: Final Top 50% of the Swarm

3.3 Constrained Problem

This is a constrained single objective optimization problem from Koziel and Michalewicz (1999) that has two variables, a single cubic objective function and is subjected to two nonlinear inequalities.

$$\begin{aligned} &\textit{Minimize} && f(x) = (x_1 - 10)^3 + (x_2 - 20)^3 \\ &\textit{Subject to} && (x_1 - 5)^2 + (x_2 - 5)^2 - 100 \geq 0 \\ & && -(x_1 - 6)^2 - (x_2 - 5)^2 + 82.81 \geq 0 \\ & && 13 \leq x_1 \leq 100\,, \quad 0 \leq x_2 \leq 100\,. \end{aligned}$$

The ratio of feasible points to sampled number of points for a 1,000,000 point random sampling is reported to be 0.000066. The optimum individual is [14.095, 0.84296] with an objective function value of -6961.81381. The first two constraints are active at the optimum.

Table 1 provides the results obtained using a swarm size of 200 flying for 100 time steps and also with a swarm size of 300 flying for 100 time steps. Five successive trials have been conducted to compute the best, worst and the average values of the objective using the present algorithm. With a swarm size of 200 and for 100 time steps, the worst, average and the best values are [-6494.0585,-6687.3350, -6916.2939] while with a swarm size of 300 flying for 100 time steps corresponding values are [-6729.7983, -6837.4347,-6883.7124]. It can be clearly observed from Table 2 that the proposed algorithm reports consistent values with much less evaluations when compared with Koziel and Michalewicz (1999). The average CPU time required for a swarm size of 200 flying for 100 time steps is 12.24s while that for a size fo 300 flying for 100 time steps is 24.04s.

Table 1: Results of the Constrained Optimization Problem

Case		1st Flight	2nd Flight	3rd Flight	4th Flight	5th Flight
Swarm Size : 200	Function Evals.	14,258	14,200	14,188	14,355	16,990
Time Steps : 100	Obj. Fn.	-6668.0268	-6596.9409	-6494.0585	-6761.3549	-6916.2939
Swarm Size : 300	Function Evals.	23,318	23,270	22,075	21,687	22,721
Time Steps : 100	Obj. Fn.	-6872.4970	-6883.7124	-6729.7983	-6828.2749	-6872.8911

Table 2: Comparison of Results

	Number of Function Evaluations	Function Value
Koziel and Michalewicz (1999)	350,000	-4236.7 (worst) -6191.2 (avg.) -6901.5 (best)
Koziel and Michalewicz (1999)	1,400,000	-5473.9 (worst) -6342.6 (avg.) -6952.1 (best)
Ray et al. (2000)	38,231 38,234 39,164	-6773.0078 -6525.8374 -6819.0391
Present Best	16,990	-6916.2939

4 Summary and Conclusions

The key to the success of the proposed swarm strategy lies with the intelligent multilevel information sharing mechanism. Pareto ranking and the generation of unique individuals are two computationally expensive operations employed within the algorithm. Such computations are meaningful for real life design optimization problems where the computation of the design objectives and constraints are equally expensive and the algorithm should be to make use of all computed information to better guide the search. The use of Pareto ranking and multilayer information sharing eliminate the need of scaling and aggregation affecting common constraint handling methods. The generation of unique individuals allows the algorithm to maintain near optimal points that are useful for sensitivity analysis. A simple generational operator has been implemented in this study but other generational operators may well be used instead.

Results of the equal optima problem clearly illustrate the capability of the algorithm to reach multiple optima simultaneously. From the unequal optima example, it is evident that the algorithm is capable to direct individuals to cluster around the global optimum instead of other local optima.

For the constrained problem, the use of Pareto ranking with intelligent constraint information sharing lead to a

faster convergence as can be seen by the drastic reduction in the number of function evaluations. The constrained and unconstrained examples clearly highlight the benefits that can be derived using the new information sharing strategy within the swarm to solve a wide variety of optimization problems.

References

1. Angeline, P. (1998). Evolutionary Optimization versus Particle Swarm Optimization: Philosophy and Performance Differences, *Proceedings of the Seventh Annual Conference on Evolutionary Programming*, March 1998.
2. Coello, C.A.C. (1999). Self-adaptive Penalties for GA based Optimization, *Proceedings of the Congress on Evolutionary Computation*, Vol. 1, pp. 573-580.
3. Deb, K. and Agrawal, R.B. (1995). Simulated Binary Crossover for Continuous Search Space, *Complex Systems*, **9**, pp. 115-148.
4. Eberhart, R. and Shi, Y. (1998). Comparison between Genetic Algorithms and Particle Swarm Optimization, *Proceedings of the Seventh Annual Conference on Evolutionary Programming*, Springer Verlag, pp. 611-618.
5. Eshelman, L.J. and Shaffer, J.D. (1993). Real Coded Genetic Algorithms and Interval Schemata, *Foundations of Genetic Algorithms II*, Morgan Kaufmann, pp. 187-202.
6. Fonseca, C.M. and Flemming, P.J. (1995). An Overview of Evolutionary Algorithms in Multiobjective Optimization, *Evolutionary Computation*, **3**(1), pp. 1-16.
7. Goldberg, D.E. and Richardson, J. (1987). Genetic Algorithms with Sharing for Multimodal Function Optimization, *Proceedings of the First International Conference on Genetic Algorithms and Their Applications*, pp. 41-49.
8. Kennedy, J., and Eberhart, R. C. (1995). Particle Swarm Optimization. Proceedings of the 1995 IEEE International Conference on Neural Networks (Perth, Australia), IEEE Service Center, Piscataway, NJ, IV, pp. 1942-1948.
9. Kennedy, J., and Eberhart, R. C. (1997). A Discrete Binary Version of the Particle Swarm Algorithm. Proceedings of the 1997 IEEE Conference on Systsems,Man and Cybernetics, Orlando, Florida, pp. 4104-4109.
10. Koziel, S. and Michalewicz, Z. (1999). Evolutionary Algorithms, Homomorphous Mappings, and Constrained Parameter Optimization, *Evolutionary Computation*, **7**(1), pp.19-44.
11. Michalewicz, Z. (1995). A Survey of Constraint Handling Techniques in Evolutionary Computation Methods, *Proceedings of the Fourth Annual Conference on Evolutionary Programming*, MIT Press, Cambridge, MA, pp. 135-155.
12. Ray, T., Tai, K. and Seow, K.C. (2000) An Evolutionary Algorithm for Constrained Optimization, *Proceedings of the Genetic and Evolutionary Computation Conference* (GECCO-2000), pp.771-777.
13. Ray, T., Liew, K.M. and Saini, P. (2001) An Intelligent Information Sharing Strategy within a Swarm for Unconstrained and Constrained Optimization Problems, Journal of Soft Computing (In Press).
14. Srinivas, N. and Deb, K. (1994). Multiobjective Optimization using Nondominated Sorting in Genetic Algorithms, *Evolutionary Computation*, **2**(3), pp. 221-248.